Designing a Quality Oceanographic Data Processing Environment

C. Paternostro[1], A. Pruessner[2], and R. Semkiw[3]

[1]Oceanic and Atmospheric Administration/ National Ocean Service / CO-OPS, 1305 East West Hwy, Silver Spring, MD 20910, Christopher.Paternostro@noaa.gov

[2]Systems Integration and Development, 15200 Shady Grove Rd, Rockville MD 20850, Armin.Pruessner@noaa.gov

[3]Systems Integration and Development, 15200 Shady Grove Rd, Rockville MD 20850, Roman.Semkiw@noaa.gov

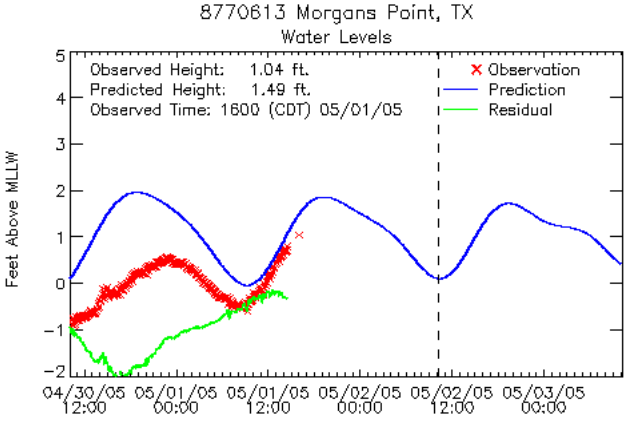
***Abstract*— Oceanographic data are increasing by data types and volume making present methods of processing and determining quality a cumbersome task. The National Oceanic and Atmospheric Administration's (NOAA) Center for Operational Oceanographic Products and Services (CO-OPS) is developing an end-to-end, state-of-the-art, public-access data management system to ingest, quality control, analyze, and disseminate water velocity and related data. The benefits include streamlining preliminary analysis, thus allowing time and resources for more in-depth investigations of the physical phenomena, increasing consistency of results between users, and improving overall data quality. The design of the system architecture follows a planned structured methodology improving the quality of the software developed. Designing a web-based modular system will allow flexibility so the system can accommodate new analyses, reports and plotting as well as allow for future data types. Well-defined algorithms will be implemented determining the quality of both the data and the analyses. This data management system will provide oceanographers the means to study water velocity data using a wide suite of mathematical and graphical tools. This will allow users to focus on the analysis results rather than the process.**

I. INTRODUCTION

The early history of oceanography and oceanic data analysis involved manual logging of data and computation. This was not only time consuming, but prone to human error. With advances in computational technology, including high-density storage devices and long term mooring capability of meters, the amount of data ingested is growing exponentially. This in turn necessitates faster turn around of data ingestion, analysis and visualization. Furthermore, quality assurance issues of the incoming data needs to be addressed.

CO-OPS has the mandate to provide the nation with accurate water level and current predictions at numerous locations along the coasts of the United States of America! Figure 1 shows an example of the valuable information provided by CO-OPS. The main mathematical tools used to provide these oceanographic predictions are harmonic decomposition and primary component analysis of time series data [1] produced by various field instruments. Present methods of analyzing

Fig. 1. Time-series of observed, predicted and residual tidal levels.



time series data utilize algorithms [2] coded in the mathematically powerful, but semantically cumbersome computer language FORTRAN.

Unfortunately, in most cases the software used for each of the phases (quality control, analysis, and visualization) involves several different software modules so that the process is disjoint and time consuming. Furthermore, the software is often developed and maintained by individual research groups with little collaboration between the groups. Whereas many other communities have publicly-available, web-based analyses and visualization tools, for example the ChipQC tool for DNA micro-array analysis in the area of bioinformatics [3] or the PAVER server for benchmarking mathematical optimization software [4], few systems exist for the oceanographic community. The EPIC system for hydrographic and time series oceanographic data [5] is undoubtedly useful but often relies on client-side desktop applications rather than being a full web-based tool. Ferret, which provides visualization and analysis of gridded oceanographic and meteorological data sets [6] has removed support for their web-based interface and is now available only as a client side application. Similarly, XTide, a harmonic tide predictor [7], also is only a client side application.

We are developing a publicly available, web-based, end-to-end data processing system for oceanographic data termed C-MIST: Currents Measurement Interface for the Study of Tides. The C-MIST system will ingest, quality control, and analyze oceanographic data. Furthermore, the system will allow visualization of data sets and user access to the data. The benefits include faster turn around time for analyses, consistency of results, and simplification of the process for the end user, ultimately allowing scientists more time to

study physical processes instead of writing and maintaining software. This paper is structured as follows: Section II describes a general oceanographic data analysis process, Section III quality software models, Section IV goes into detailed system design, illustrating several key features of a quality oceanographic data analysis system and in Section V we draw conclusions.

II. OCEANOGRAPHIC DATA ANALYSIS

Oceanographic data analysis is a broad field thus we will not focus on all issues involved. The following highlight key areas the software system will address. In particular, the system needs to handle:

1) Quality control (QC) and data ingestion
2) Data analysis
3) Data visualization and report generation

A. Quality Control and Data Ingestion

The data usually arrives at Headquarters in a binary, vendor-specific format and needs to be converted into columnar ASCII format before any analysis may proceed. Although all raw data will be ingested into the system database, verified data must first be quality controlled. This includes checking for missing data records, instrument pitch and roll levels during the duration of the measurement, instrument heading, water velocity, and percentage of good pings of the instrument, just to mention a few of the quality checks. Data not satisfying these will be flagged appropriately. Data passing these checks are available to proceed with further analysis. Furthermore, data may be converted as needed for uniformity of units and corrections may be applied, *e.g.*, correcting water velocities to true north.

B. Data Analysis

The general notion of the data analysis phase is to help oceanographers analyze large data sets and be able to extrapolate future tidal forces based on the available data. Arguably, the most common technique for processing of sequential data is time-series analysis. The idea is to define the periodic data in terms of a finite number of dominant period functions [3], [1]. At CO-OPS we make use of various methods for fitting a finite sequence of cosine terms (*i.e.*, determining amplitude and phase of each term) to the observed data, depending on the amount of data available.

These include:

1) Least squares harmonic analysis (more than 180 days of data)
2) 29 day Fourier harmonic analysis (between 29 and 180 days of data)
3) 15 day Fourier harmonic analysis (15-29 days of data)

The resulting constituents (amplitude and phase coefficients) allow for modeling and prediction of future tidal forces. In particular, using the constituent information obtained, we make tidal predictions for a whole year, every year, for more than 2,700 water velocity stations[8].

Fig. 2. Time series of Cook Inlet station west of Cairn Point on July 22-26, 2003. Shows the observed, predicted and residual currents along the principle axis of flow.

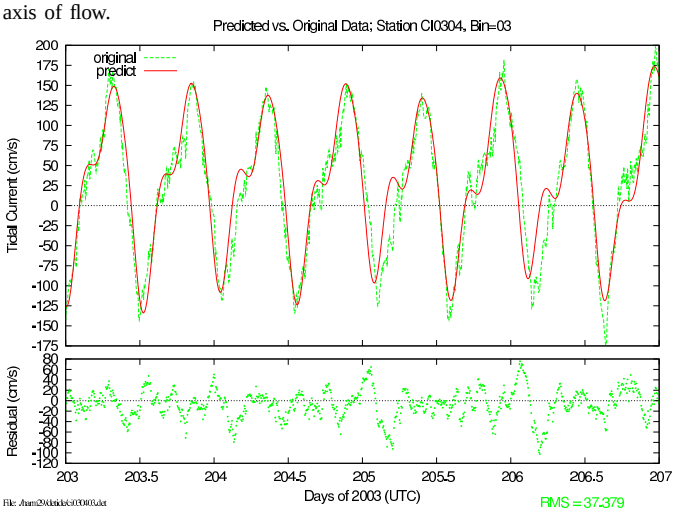
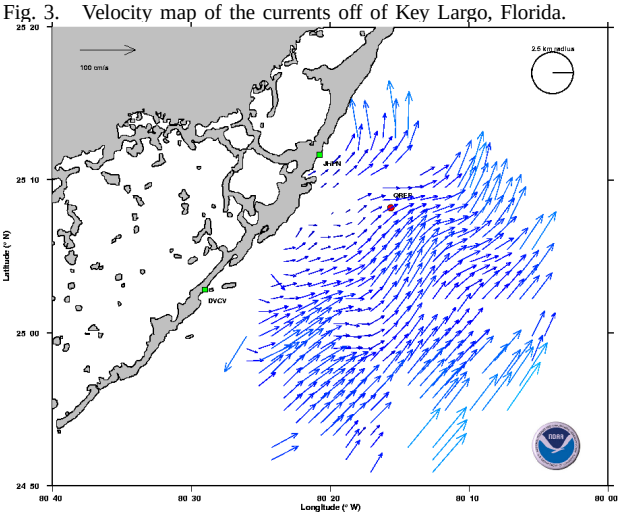


Fig. 3. Velocity map of the currents off of Key Largo, Florida.



After the mandatory data analysis, the C-MIST system will have the added capability of conducting quality control routines upon the results. This assures that the analysis is correct by providing statistics and calculations to the user for verification of the results. The data can then be used for alternate computations beyond the analyses presently available.

C. Data Visualization and Report Generation

C-MIST will allow the user to generate a wide variety of plots, such as time series plots of the observed, predicted and residual tide currents over a given time period (see Figure 2, vector plots (see Figure 3), as well as animations showing information over a wider time period. In addition, users have access to standard as well as customized reports generated on-the-fly by the system.

III. QUALITY SOFTWARE MODELS

Quality software generally follows a structured approach based on standard quality models. We chose the Capability Maturity Model (CMM) [9] to guide us in our development

Fig. 4. Linear method to develop quality software.

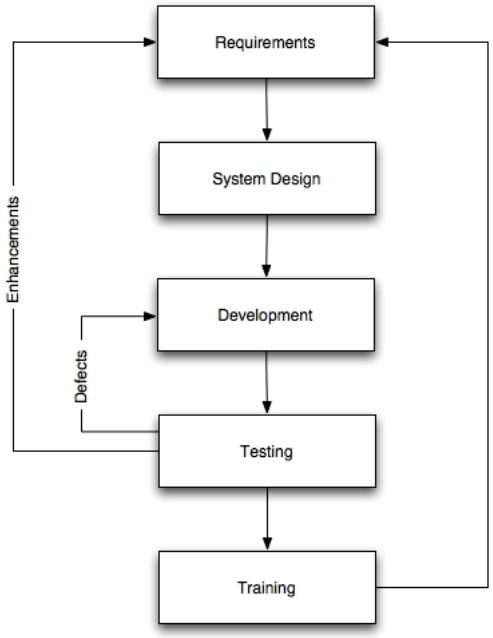
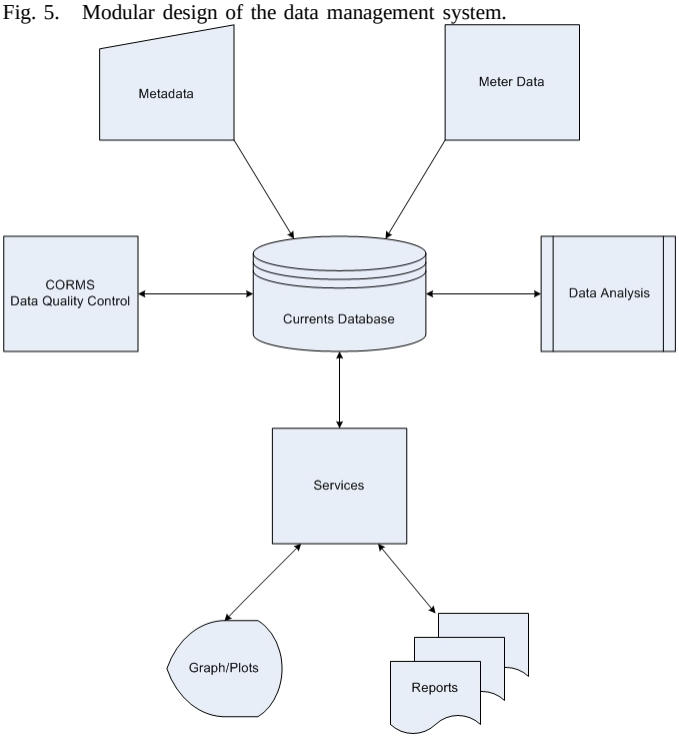


Fig. 5. Modular design of the data management system.



efforts. The linear model allows for a structured approach and encapsulates quality control measures at each step of the process (see Figure 4). The idea is to uncover defects as early as possible in the process lifecycle. While initial requirements and system design certainly require more resources initially, the long term benefit is clear: fewer timely defects, lower maintenance costs, as well as improved design and functionality.

Furthermore, test cases are constructed at each phase a bug is found, thus eliminating the possibility of the same defect recurring. The notion is that quality assurance takes place at each phase of the process, not just in the final testing phase. This design philosophy improves the final product by making quality an integral building block of the software engineering process. Another important issue is the modular design of the system (see Figure. 5). By modular design we mean that the system is constructed of individual components which interact with each other via standard input/output libraries. This uniform design allows for simple addition of new modules. For example, suppose a new plotting routine needs to be implemented. The existing infrastructure greatly simplifies the addition by allowing black-box type of interfacing. Again, this greatly speeds up development, improves quality assurance (since the input-output interface has already been tested previously), and decreases overall costs.

IV. SPECIFIC IMPLEMENTATION ISSUES

Many implementation issues need to be considered, but for brevity we will highlight only some of the important ones below. These include, but are not limited to

- Exception handling
- User friendliness
- Load testing
- Documentation and coding conventions
- Security

A. *Exception Handling*

Unfortunately with any software system, inevitably, exceptions will occur. By exceptions, we mean unexpected software behavior. Note that this is in contrast to software failure, which we generally refer to in the event of catastrophic behavior where no user-friendly information is returned regarding the exception. In other words, in part, the difference between "good" and "bad" software is in the way exceptions are handled. Good exception handling means that the user will always receive an informative message indicating the anomaly, which can be relayed to the system administrator for further debugging. The C-MIST system will have built-in exception handling mechanisms to cope with unanticipated behavior. Furthermore, events are logged so that in case of exceptions, useful debugging information is readily available. This pro-active approach to exceptions (in contrast to a reactive approach when a bug is encountered) results in faster turn around time for identifying and fixing defects.

B. *User Friendliness*

A common measure of software quality is if the end-user is satisfied with the product. This may seem like a non-rigorous validation metric, but ultimately the product is designed for the end user. In other words, even if the software passes all requirements, if the end user is not satisfied with the product, then the software QA measures have failed. One issue often overlooked is the user-friendliness and intuitiveness of a system. Too often software systems are designed with extensive functionality, but lack clean interfaces and easy-to-use web menu options. Having end users involved early in the

requirements and prototype phases of the software lifecycle minimizes the chance of surprises later on in the lifecycle and results in an end product closely aligned to the requirements of the end users.

C. Load Testing

Load testing refers to modeling the expected usage of a software program by simulating multiple users accessing the program's services simultaneously. This includes testing system response to peak or unusually high loads. At NOAA, the latter is important because user load is highly dependent on weather patterns or events. For example, the hurricane season caused a four-fold increase in the number of users to the CO-OPS tides website. See Figure 5 for details. Any system must be able to respond to spikes in user loads and have exception handling capabilities in case of too great a user load.

D. Documentation and Coding Conventions

Documentation is often left as a tedious task at the end of the software development process. Good documentation is key to user comprehension of system features and usage. Furthermore, documentation within the software code itself is important in maintenance and re-engineering of modules. As a general rule, software often lasts longer than its intended lifespan and multiple developers usually work on code during this time. Without consistent documentation throughout, the time involved to do any modifications is increased. Another issue is code naming conventions: developers should follow a consistent scheme for variables (local and global), classes, and modules to simplify reading comprehension.

Fig. 5 Unexpectedly high loads on the CO-OPS website in September 2004 due to the hurricane season.

E. Security

Increasingly, software security is becoming a more and more important consideration in the development process, with many requirements specifying security within the basic requirements. Security needs to address issues regarding

- Maintaining database integrity
- Maintaining server integrity
- Maintaining web page integrity

The first aspect requires security regarding the data - in particular gaining unauthorized privileged (root) access to the database. This may result in users being able to update or even delete database records. Server integrity refers to users not being able to gain privileged access to the web server itself. Otherwise, a malicious user may be able to delete files on the server, update content (web page integrity), or crash the server itself. By designing and developing software with security in mind from the ground up, long term maintenance costs for the software can be reduced and a more stabled system developed.

V. CONCLUSIONS

We have outlined a state-of-the-art end-to-end public oceanographic data analysis system, which improves on the current state-of-the-art. The C-MIST system modular design will allow for easy maintainability and extension. Furthermore, the web-based platform-independent design allows accessibility from various locations and is open to the public. The system is designed to be able to handle larger data sets that have higher spatial and temporal resolution. The data management system will provide oceanographers the means to study water velocity data using a wide suite of mathematical and graphical tools allowing them to create more products rather than spending time altering data sets.

REFERENCES

[1] P. Shureman, *Manual of Harmonic Analysis and Prediction of Tides*. No. Special Publication No. 98, US Department of Commerce, Coast and Geodetic Survey, revised edition ed., 1940.

[2] C. Zervas, "Tidal Current Analysis Procedures and Associated Computer Programs," Technical Report NOS CO-OPS 0021, NOAA, US Department of Commerce, Silver Spring, MD, 1999.

[3] W.J. Emery and R. E. Thompson, *Data Analysis Methods in Physical Oceanography*. Elsevier BV, second and revised edition ed., 2004.

[4] H.D. Mittelmann and A. Pruessner, "A Server for Automated Performance Analysis and Benchmarking of Optimization Software, Optimization Methods and Software." 2005.

[5] D.W. Denbo, N.N. Soreide, W.H. Spillane, and W.H. Zhu, "EPIC: Providing World Wide Web access to oceanographic observations.," in *15th International Conference on Interactive Information and Processing Systems (IIPS) for Meteorology, Oceanography, and Hydrology*, (Dallas, TX), AMS, 1999.

[6] "Ferret, Interactive computer visualization and analysis environment.," tech. rep., NOAA/Pacific Marine Environmental Laboratory, Seattle, WA, 26 Nov. 2003.

[7] David Flater, "XTide, A package providing tide and current predictions in a wide variety of formats." http://www.flaterco.com/xtide/, 2005.

[8] "Tidal Current Tables 2005," Data Table NOSPBTCTATCSTN5, Department of Commerce/NOAA, Silver Spring, MD, 2004.

[9] Software Engineering Institute, *The Capability Maturity Model: Guidelines for Improving the Software Process*. Reading, MA: Addison-Wesley, 1994.